Quality of Service Renegotiations*

Werner Almesberger †, Leena Chandran-Wadia †, Silvia Giordano †,
Jean-Yves Le Boudec †, Rolf Schmid ‡

Abstract: Giving users simple access to QoS selection capability of ATM networks is important for the future of integrated services networks. We describe the first instance of an application in which users had the capacity to tune ATM traffic parameters with a button, at run time. To achieve this we implemented the necessary signaling extension on ATM switches as well as on end-systems. Furthermore, we extended *Arequipa* (Application REQested IP over ATM; a mechanism that enables applications to request direct ATM connections for their exclusive use, and to directly control their traffic parameters) to support modification, and modified the popular Mbone tool *Vic* (VIdeo Conferencing) to use *Arequipa* to request and modify ATM bandwidth at will. We describe how *Vic* accomplished this over the ATM WAN of SWISSCOM, transferring live video from Lausanne to Basel over switched, renegotiable ATM connections.

1 Introduction

ATM networks [2, 3] offer a rich set of features to support guaranteed quality of service (QoS), including several traffic and QoS[1] parameters which can be set by the user. Despite this rich structure, the main body of applications today still use ATM only via the IP layer, and do not benefit from the possibilities for QoS selection offered by ATM [4].

We have been studying ATM networks with a view to making the QoS selection capability visible to the user, and to do this in as simple a way as possible. Towards this end we have developed a mechanism, *Arequipa*, by which IP applications can have the option of selecting QoS parameters. Among the many choices for the latter we believe that, from the users' point of view, it is appropriate to restrict that choice to just the bandwidth, or ATM peak cell rate (PCR).

*work done in collaboration with the ACTS EXPERT project - AC094 [1]

†ICA (Institute for computer Communications and Applications), EPFL (Swiss Federal Institute of Technology), Lausanne, Switzerland

‡ASCOM Tech. Ltd, Bern, Switzerland

[1]We use QoS to refer to quality of service in general but also sometimes to ATM connection QoS parameters. The meaning will always be clear from the context.

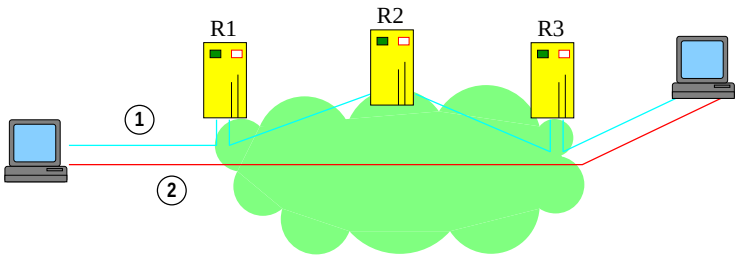

Figure 1: An *Arequipa* connection on ATM network (2) which bypasses intermediate IP routers (on IP network (1)) completely

The ATM Forum and the ITU [5, 6] define a large number of ATM connection types, ranging from constant bit rate (CBR) to Available bit rate (ABR). In the work reported here, we focus on the use of *renegotiable CBR* connections, because they provide a simple means to offer a visible quality of service, under explicit control from the end-user. Renegotiable CBR connections are connections with a maximum peak cell rate, which can be modified by the user at any time (see Section 2).

Arequipa or Application REQuested IP over ATM [7, 8] is a method for providing the quality of service of ATM to TCP/IP applications, assuming end-to-end ATM connectivity exists. The *Arequipa* mechanism does not require any changes in the network but two changes need to be made at the end systems. First, in order to implement *Arequipa*, some changes need to be made to the TCP/IP protocol stack (on the end systems). Second, applications using *Arequipa* to obtain QoS need to be modified slightly to make use of our simple extension to the socket interface. The latter consists of just four calls for setting up, tearing down and modifying the traffic parameters of connections. The applications then make use of the end-to-end ATM connectivity and *Arequipa* to set up a direct switched virtual channel connection (SVC) from the sender to the receiver application (Fig. 1).

We have partially implemented UNI4.0 signaling and the Q.2963.1 connection modification capability on the end systems and on switches. Simultaneously we have extended the *Arequipa* mechanism and API to include the connection modification capability. With these an application can now modify the QoS parameters at run time, *after* connection setup.

We demonstrate this in the case of the Mbone VIdeo Conferencing tool *Vic* which we have modified to be Arequipa-capable. *Vic* was used [9] to video conference with QoS, in point-to-point mode, between Basel and Lausanne and also between Zürich and Lausanne. We describe details of this demo which was done over the ATM WAN of SWISSCOM. To our knowledge this was the first time any application had the ability to tune ATM traffic parameters at run time.

Arequipa only works in situations where end-to-end ATM connectivity exists, but this assumption is not unrealistic in Europe where many countries have

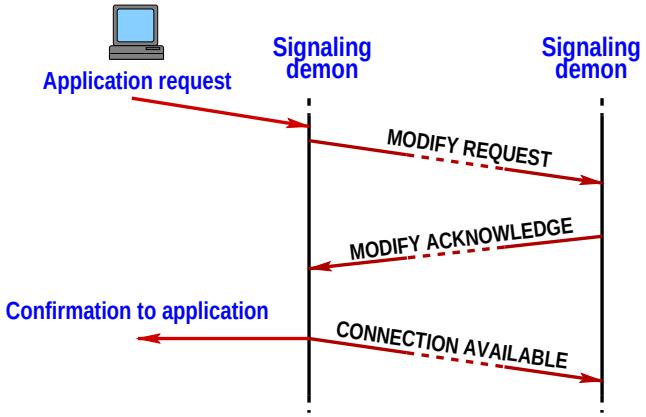

Figure 2: Message sequence during bandwidth renegotiation

extensive ATM already deployed, not just in the backbones but also in the LANs.

The following section describes the main features of UNI4.0 signaling and Q.2963.1 connection modification capabilities. Sections 3 and 4 describe *Vic* and *Arequipa* respectively, while section 5 describe details of the demo.

2 QoS Renegotiations

We call a CBR connection *renegotiable* if its traffic parameters can be modified after the connection is set up. A renegotiable CBR is different from a negotiable CBR connection where traffic parameters can only negotiate at call setup time. Renegotiation is currently defined only as the connection modification capability in the ITU recommendation Q.2963.1. The only connection characteristic that can be modified according to Q.2963.1 is the peak cell rate (PCR).

Fig. 2 illustrates the signaling messages that are exchanged during bandwidth renegotiation. When an application requests a modification, a MODIFY REQUEST is sent out to the called user provided local resources are available. The called user checks if resources are available at its side and if so acknowledges the modification. There is once again a check of local resources at the calling user before the modification is definitely accepted. Then a confirmation is sent to the application and the peak cell rate is changed. A renegotiation requesting a rate increase may fail, in which case the peak cell rate prior to the renegotiation continues to apply.

Connection modification is going to be an important capability in future commercial broadband networks, where users will want to control the speed and quality of the data stream because they will be charged for network usage (e.g. bandwidth, time). Modification is especially suited for interactive multimedia applications where bandwidth usage is likely to vary considerably over time.

2.1 Implementation

ATM signaling support, including *Arequipa* and the connection modification capability at the end systems has been implemented for PC's running the Linux operating system [10]. Signaling and connection modification has also been implemented on the ATMLightRing switch of ASCOM. The ATM on Linux distribution containing full source code for kernel changes, system programs and test application is available publicly [10]. *Arequipa* (see section 4) is also part of the ATM on Linux distribution [11] of ICA. For details on *Arequipa* see [12]

3 VIC

The Mbone [13, 14] VIdeo Conferencing tool *Vic* [15] has a modular architecture characterised, among other things, by network layer independence and an extensible user interface. We have used these two features in order to enable *Vic* to use *Arequipa*. The user interface has been modified to include elements for *Arequipa* control and a new network module has been added to set up the appropriate *Arequipa* connections when necessary.

The *Vic* distribution [16] contains three separate network modules for normal IP, native ATM and RTIP (Real-Time IP). Only one of these can be linked at a time, in the current version of *Vic*. The new, fourth, "*Arequipa*" network module we have added defaults to the normal IP module if *Arequipa* fails for any reason. As mentioned earlier, *Vic* can then make use of the end-to-end ATM connectivity and the "*Arequipa*" network module to set up a direct SVC, with prescribed QoS, for video transfer from sender to receiver.

In this work we only use *Vic* in the point-to-point mode using standard unicast IP addresses, even though *Vic* was primarily intended as a multi-party conferencing application on the Mbone. For such point-to-point video transfer, both the sender and receiver of video need to know each others' IP address and to agree on a port number. As with the normal IP network module, this is settled out of band.

3.1 User Interface

The main window of *Vic* containing thumbnail views of the outgoing (loopback) video as well as the incoming video is shown in Fig. 3. Each thumbnail picture is accompanied by identification text, frame and bit rate statistics, and a loss indicator (in parenthesis). The latter is inferred from sequence numbers of the incoming packets. Packets are lost either due to network drops or due to local socket buffer overflows resulting from CPU saturation. Fig. 3 is taken in Basel, using the new "*Arequipa*" network module but before starting *Arequipa*. It shows a loss rate of the incoming video from Lausanne (over the Internet) to be well over 50%, despite using relatively low bit rates.

Details of the *Vic* control panel (obtained by clicking on the menu button in the main window) are shown in Fig. 4. We have extended the menu by adding

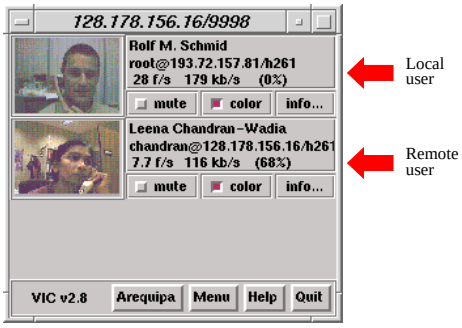

Figure 3: Main window of *Arequipa*capable *Vic*

a section on top for *Arequipa* control. By design only the sender of video can initiate an *Arequipa* session, by selecting the service category and the bandwidth (converted internally into PCR) and clicking on the *Arequipa*button [2].

The only user specifiable traffic parameter is the bandwidth and CBR/UBR the only choice of service categories. Once an *Arequipa* connection is in place, it it not possible to change the service category (e.g. from CBR to UBR), but only to change the traffic parameter. Each time the content of the bandwidth entry is modified by the user, the PCR of the associated SVC is modified automatically via a call to `arequipa_renegotiate` (see section 4.2).

The capture (and encode) frame and bit rates are displayed above the QoS settings. The close button only closes the *Arequipa* connection and video transmission continues via the normal IP path. As in the unmodified *Vic*, the release button has to be used to stop video transmission.

Arequipa can be started any time during (IP) video transmission, e.g. when picture quality deteriorates. If normal IP transmission has not already begun when *Arequipa* is started then *Vic* first starts it before switching over to *Arequipa*. Then, if *Arequipa* fails for any reason, transmission reverts to the normal Internet path.

3.2 Networking

As described in detail in [15] *Vic* uses the Real-time Transport Protocol, RTP [17], which is realised completely within *Vic* itself. RTP is divided into two components: the data delivery protocol and the control protocol RTCP. The former handles the actual media transport and the latter manages control information like sender identification, receiver feedback and cross-media synchronisation.

The new network module we have added contains, apart from the normal IP module, procedures for the exchange of ATM addresses, for opening, closing and renegotiating *Arequipa* connections. Note that in order to make the connection, the `arequipa_preset` (see section 4.2) call requires the knowledge of the ATM

[2] note that the *Arequipa* mechanism itself does not carry any constraint as to which side can initiate *Arequipa*.

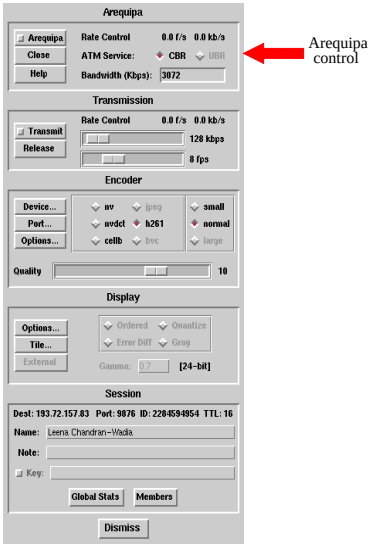

Figure 4: Control panel of *Arequipa* capable *Vic*

address of the peer machine. So far *Arequipa* connections have only been set up for the data channel. The control traffic, being low volume, does not really need an *Arequipa* connection.

4 Arequipa

4.1 Overview of Arequipa

Arequipa is a method for allowing IP applications running on hosts that have direct ATM connectivity to send traffic on an exclusive ATM connection (SVC). The QoS is guaranteed by the fact that each of these SVCs is used exclusively for one IP flow identified by a pair of sockets (eg. a TCP connection or a UDP stream).

Arequipa is completely transparent to both the ATM and the IP networks, therefore it scales very well for large networks and does not have to be available to all the networks elements. Moreover, *Arequipa* capable applications coexist with "non arequipa" ones, thus no modifications are necessary to the latter, and they can still communicate with the original IP stack.

On the other hand, as foreseeable, it requires two important changes at the end-systems. Some changes need to be made to the TCP/IP stacks at the end systems (discussed in detail in [8]). The other modification needed is obviously to the application, to make it QoS aware. This means that the application needs to be modified to make use of the *Arequipa* socket extensions. It is important to note that if an application is to be QoS aware at all, some code needs to be added somewhere, either in the applications themselves or in some proxies or

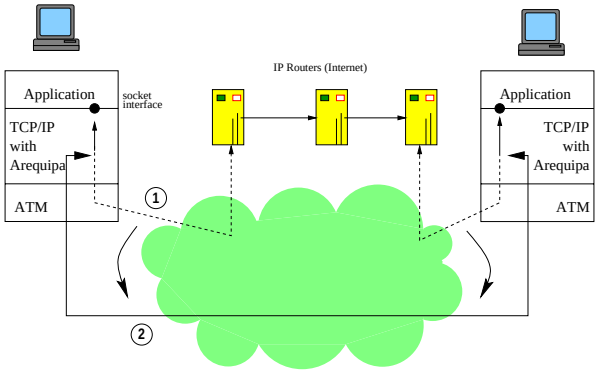

Figure 5: *Arequipa* capable applications: data transmission is switched from the default IP path (1) to ATM (2).

gateways. We argue that modifying *Vic* to use *Arequipa* is not more complex than modifying it for RSVP [18, 19], and certainly less so than modifying it for native ATM [16].

4.2 Functionality

Arequipa adds four simple new functionalities at the socket layer. Applications need to be modified to use just the following four calls in order to set-up, tear-down or modify their ATM SVCs.

- `arequipa_preset(socket, atmaddr, qos)`: sets up a bidirectional VCC, symmetric or asymmetric, with a given ATM service and QoS.
- `arequipa_expect(socket, {true, false})`: preparing a socket to use an incoming *Arequipa* connection for all its outgoing traffic.
- `arequipa_close(socket)`: implicit or explicit closing of *Arequipa* connections.
- `arequipa_renegotiate(socket, newqos)`: renegotiation of existing *Arequipa* connections.

5 Demonstration over an ATM WAN

Vic was used to transfer live video in several demos from Lausanne to Basel and Zürich over SWISSCOM's public ATM network. The first demo was conducted on 24th October and the second during the REFORM open-day on the 22nd of January 1998 in Basel.

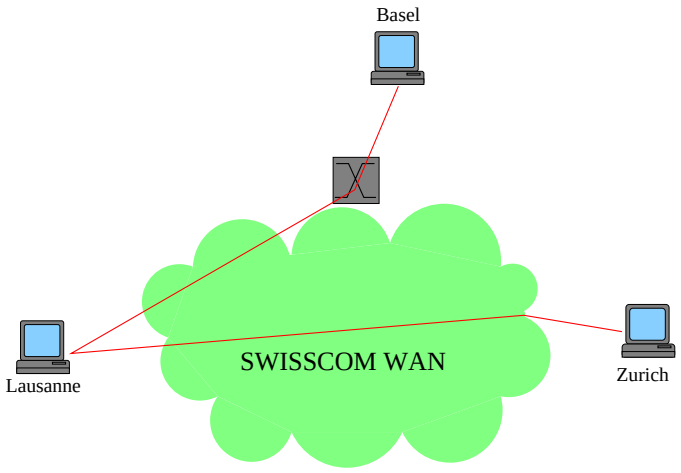

Figure 6: Topology of the network used in the demo

5.1 Setup

The demonstration network consisted of two ATM equipped Linux terminals located in Lausanne and Basel, connected to a (two node) ATMLightRing System at the EXPERT testbed in Basel. The two sites were interconnected by the VP connection provided by SWISSCOM's public network. The demonstration network between Basel and Lausanne is illustrated in Fig. 6. None of the intermediate switches except the ATMLightRing supported the UNI4.0 and the Q.2963.1 signaling capabilities. The VP connection from SWISSCOM provided seamless connectivity between the end systems and the ATMLightRing.

5.2 Observations

As shown by the loss indicator of the incoming video in Fig. 3, the loss rates over the Internet can often be well over 50% even at relatively low bandwidth. Switching over to *Arequipa* at such times results in a dramatic change in the quality of the video, because then network losses in congested routers no longer occur. At low bandwidths the loss rate drops to zero immediately. At much higher bandwidths losses make their appearance once again, this time due to CPU saturation. The threshold at which losses appear depends on the system configuration, but also on the type of encoding being used.

6 Discussion and Summary

We have shown that in areas where end-to-end ATM connectivity already exists, there is a relatively simple way for IP applications to use the QoS of ATM, namely, by using *Arequipa* and its simple API. We demonstrated this in the

case of *Vic* which also became the first instance of an application to tune bandwidth at run time.

With end-to-end ATM connectivity, there is an alternate way of providing hard QoS guarantees to applications. This is to specifically enable them to use ATM, resulting in a so-called *native* ATM application. In the case of *Vic*, this has been done and *Vic* can run on ATM using the FORE SPANS API. In general, the effort involved in converting an IP application into a native ATM one can often be significant. In the case of *Vic* this was certainly so. The native ATM network module is completely different from the normal IP module whereas the *Arequipa* network module is merely an extension of it, using the same sockets and so on.

The IETF way of providing quality of service (QoS) guarantees to applications in IP networks today is to use the ReSerVation Protocol (RSVP) [18]. As with *Arequipa* and with native ATM, applications need to be modified in order to make them QoS sensitive. *Vic* [15] has also been RSVP enabled [19] and can provide soft guarantees on the QoS depending on the extent of RSVP deployment on the intermediate routers. With the work reported here, we believe to have demonstrated that presenting explicit quality of service to the application and their users is indeed simple and can be deployed as soon as networks exist which support reservations.

7 Acknowledgements

We would like to thank several people who contributed to the success of this work in different ways: Rainer Burki, Alan Cox, Mireille Goud, Paco Hope, Paul Hurley, Jörg Liebherr, Philippe Oechslin, Koji Okamura, Anne Possoz and Petar Vrdoljak. We would also like to thank SWISSCOM for the use of the WAN and ASPA Basel for support and use of the EXPERT testbed.

References

[1] EXPERT Consortium, *EXPERT AC094 home page*, 1997. http://www.elec.qmw.ac.uk/expert/intro.html.

[2] J.-Y. Le Boudec, "The Asynchronous Transfer Mode : A Tutorial," *Computer Networks and ISDN Systems*, vol. 24 (4), pp. 279–309, May 1992.

[3] A. Alles, "Interworking with ATM," *InterOp proceedings*, 1995.

[4] S. Giordano, R. Schmid, R. Beeler, H. Flinck, J.-Y. Le Boudec, "IP and ATM - current evolution for integrated services ," Technical Report 98/2, DI-EPFL, CH-1015 Lausanne, Switzerland, January 1998.

[5] The ATM Forum, *ATM User-Network Interface (UNI) Signalling Specification, Version 4.0*, 1996. ftp://ftp.atmforum.com/pub/approved-specs/af-sig-0061.000.ps.

[6] ITU Telecommunication Standardization Sector - Study group 13, *ITU Recommandation Q.2931, Broadband Integrated Services Digital Network (B-ISDN) – Digital subscriber signalling system no. 2 (DSS 2) – User-network interface (UNI) – Layer 3 specification for basic call/connection control*, 1995.

[7] W. Almesberger, J-Y. Le Boudec, Ph. Oechslin, *RFC2170: Application REQuested IP over ATM (AREQUIPA)*. IETF, July 1997.

[8] W. Almesberger, J-Y. Le Boudec, Ph. Oechslin, "Arequipa: TCP/IP over ATM with QoS ... for the impatient," Technical Report 97/225, DI-EPFL, CH-1015 Lausanne, Switzerland, January 1997. ftp://lrcftp.epfl.ch/pub/arequipa/impatient.ps.gz.

[9] LRC-EPFL, L. Chandran (Editor), *Web over ATM: Intermediate Report*, 1997. http://lrcwww.epfl.ch/WebOverATM/finaldemo.html.

[10] W. Almesberger, *ATM on Linux Distribution*. EPFL, 1997. ftp://lrcftp.epfl.ch/pub/linux/atm/dist.

[11] W. Almesberger, *ATM on Linux*. EPFL, 1996. ftp://lrcftp.epfl.ch/pub/linux/atm/papers/atm_on_linux.ps.gz.

[12] W. Almesberger, "Arequipa: Design and Implementation," Technical Report 96/213, DI-EPFL, CH-1015 Lausanne, Switzerland, November 1996.

[13] H. Eriksson, "Mbone: The multicast backbone," *Commun. of the ACM*, pp. 54–60, Aug. 1994.

[14] V. Kumar, *MBone: Interactive Multimedia on the Internet*. Indianapolis, IN: New Riders, 1996.

[15] S. McCanne and V. Jacobson, "vic: A flexible framework for packet video," *ACM Multimedia*, 1995.

[16] Vic-Distribution, *VIdeo Conferencing*, 1997. ftp://ee.lbl.gov/conferencing/vic.

[17] H. Schulzrinne, S. Casner, R. Frederick, and V. Jacobson, *RFC1889: RTP: A Transport Protocol for Real-Time Applications*. IETF, 1996.

[18] R. Braden and L. Zhang and S. Berson and S. Herzog and S. Jamin, *RFC2205: Resource ReSerVation Protocol (RSVP) - Version 1 Functional Specification*. IETF, September 1997.

[19] S. Berson, "RSVP enabled Vic," , 1996. ftp://ftp.isi.edu/rsvp/release.